UNIVERSIDAD AUTONOMA DE MADRID

ESCUELA POLITECNICA SUPERIOR

Grado en Ingeniería Informática

TRABAJO FIN DE GRADO

Módulo de edición para un tutor online de C

Gonzalo Hernández Muñoz
Tutor: Alejandro Sierra Urrecho

FEBRERO 2017

Módulo de edición para un tutor online de C

AUTOR: Gonzalo Hernández Muñoz
TUTOR: Alejandro Sierra Urrecho

Dpto. Ingeniería informática
Escuela Politécnica Superior
Universidad Autónoma de Madrid
Febrero de 2017

Resumen (castellano)

Los entornos de aprendizaje virtual son una herramienta en continuo crecimiento hoy en día. Son un complemento ideal al aprendizaje tradicional y presencial y, sin duda, una de las mayores ventajas de este sistema es la posibilidad que otorga al alumno para imponerse su propio ritmo de trabajo. Esta personalización e interactividad fomentan que los estudiantes sean los responsables de su propio aprendizaje.

Para que esto sea posible estas plataformas deben permitir un acceso completo y estructurado a los contenidos que se desean impartir, por ello la gestión de contenidos tiene un peso importante en el desarrollo de este tipo de aplicaciones.

Mediante la realización de este Trabajo de fin de Grado, se ha conseguido añadir un módulo extra a una aplicación de enseñanza online. La aplicación en cuestión es un curso de introducción a la programación en lenguaje C, que proporciona distintos contenidos y actividades dirigidas a alumnos de estudios de Ingeniería informática. Actualmente la aplicación se encuentra en funcionamiento y algunas secciones ya se encuentran plenamente funcionales, como la gestión de los usuarios, o el módulo de visualización de estadísticas sobre el rendimiento de los alumnos. Otras, en cambio, aún no han sido desarrolladas y se espera que se añadan en un futuro sin afectar al correcto funcionamiento de la aplicación.

En concreto, la parte que atañe a este Trabajo de Fin de Grado se corresponde con la edición y la ampliación del temario y actividades incluidos en la aplicación. Los contenidos se encuentran divididos en distintas unidades con un nivel de dificultad progresivamente más elevado. Cada una de estas unidades pone a disposición del alumno una serie de explicaciones teóricas acompañadas de ejercicios a los cuales el alumno debe responder y son corregidos de manera automática en la aplicación. También se incluye en cada una de las páginas un código en lenguaje C que es posible compilar, ejecutar y corregir, permitiendo al alumno realizar las modificaciones oportunas sobre el mismo.

El objetivo principal de este Trabajo de Fin de Grado consiste en facilitar el trabajo a la hora de añadir contenidos en la aplicación mediante el desarrollo de un nuevo módulo independiente, de modo que el personal docente que la usa pueda realizar estas modificaciones de contenido de una manera eficiente. Esta nueva funcionalidad permitirá a las personas sin conocimientos de programación poder realizar estas tareas a través de un formulario estándar, ya que hasta el momento la manera de agregar o editar estas unidades era realizar modificaciones directamente en el código de la aplicación lo cual es un trabajo laborioso. Otro aspecto importante y esencial de este Trabajo de Fin de Grado es que todo el desarrollo de este módulo se lleve a cabo afectando mínimamente al resto de la aplicación, debe mantenerse funcional en todo momento, y a la vez, el nuevo módulo ha de ser compatible con los actuales que ya se encuentran operativos siendo también un requisito imprescindible que dicho modulo sea ampliable y/o modificable fácilmente en caso de ser necesario.

Palabras clave (castellano)

e-learning, Gestión de contenidos, PHP, Laravel, MySQL, Sistema de gestión de aprendizaje (LMS).

Abstract (English)

Virtual learning environments are tools in constant growth and development. They are the ideal complement to traditional face-to-face learning methods. One of the greatest advantages of this kind of system is that it allows the students to set their own pace of work. This personalization and interactivity encourage the student to be responsible for their own learning process. For this to be possible these platforms should grant full and structured access to the contents that will be taught, and that is why content management has great importance in the development of these applications.

The development of this project has made it possible to add new modules to an existing online learning application.

The application is an online training course intended for learning how to program in the C programming language. It provides a variety of different contents areas and activities aimed to students of computer science engineering. At the present time the application is already online and working. Some parts of it, like user management or the stats module are totally functional while other parts are still under development or the development has not yet begun. It is expected that new functionality will be added to the application in the near future without affecting the already working modules and without affecting the proper functioning of the website.

Specifically the part that concerns the work explained in this document is that which is related to the edition and addition of new units and exercises included in the application. The contents are divided into units with an increasing difficulty level. Each of these units makes available to the user a series of contents that includes a brief explanation sometimes accompanied by exercises which the user needs to complete and which are automatically corrected in the application. It is also included in each page a section with a C language code that it is possible to modify, compile and execute by the user, allowing the students to test the learned skills.

The main goal of this project is to make the task of adding or editing content in the application easier, so the teaching staff can do it more efficiently and without worrying about the errors that can occur when adding new content by editing the source code of each unit and also allowing people without a computer science background to do this task with a simple form. There will not be the need to edit the source code of the application, saving that laborious work. Another essential aspect of the project is to keep the application working while the new module is developing and at the same time, the new module has to be compatible with the old ones that are already operational and fully working. And finally the new parts that this project includes have to be able to expand or modify easily.

Keywords (inglés)

e-learning, Content management, PHP, Laravel, MySQL, Learning Management System (LMS)

Agradecimientos

Hace unos años comencé mis estudios de Ingeniería informática siendo una persona completamente diferente a la que soy hoy. Todas las decisiones tomadas a lo largo de estos años me han llevado hasta este instante concreto, los buenos y malos momentos, las risas y las lágrimas, el trabajo duro y las épocas de descanso, pero en ese camino siempre he podido contar con la ayuda de las mismas personas, es por ello que me gustaría agradecer a todos mis amigos que siempre han estos ahí conmigo en esos pequeños instantes, a mi familia, mis padres Celso y Maribel, y mi hermana Elisa. Pero sobretodo me gustaría dedicar este trabajo de fin de grado a una persona que desde que llegó a mi vida, su apoyo, fuerza y entusiasmo por la enseñanza me han ayudado a completar tanto este trabajo como cualquier cosa que me proponga. Te lo dedico a ti Claire.

Me gustaría terminar este agradecimiento con una pequeña cita sobre el discurso de graduación que George Saunders dio en la universidad de Siracusa, Nueva york, en 2013.

"And so, a prediction, and my heartfelt wish for you: as you get older, your self will diminish and you will grow in love. YOU will gradually be replaced by LOVE. If you have kids, that will be a huge moment in your process of self-diminishment. You really won't care what happens to YOU, as long as they benefit. That's one reason your parents are so proud and happy today. One of their fondest dreams has come true: you have accomplished something difficult and tangible that has enlarged you as a person and will make your life better, from here on in, forever."

George Saunders – May 11, 2013

Y es este precisamente el cambio que todos debemos experimentar y que yo he experimentado, ser menos egoístas conforme maduremos, para que al final, seamos solo amor.

INDICE DE CONTENIDOS

INDICE DE FIGURAS

1 Introducción

1.1 Motivación

El mundo actual, y con él, el sector de la enseñanza están actualmente viviendo una revolución tecnológica. Cada día surgen todo tipo de cursos online y las universidades complementan las clases impartidas en el aula con cursos interactivos online que los alumnos pueden seguir desde sus propias casas. El número de estudiantes universitarios que demandan cursos online ha crecido en los últimos años (1), pero no solo aumenta la cantidad, sino que la calidad está aumentando a la par.

La enseñanza online proporciona numerosas ventajas, otorga flexibilidad al alumno, permite una mayor participación de los alumnos, ya sea mediante feedback dentro de la misma aplicación, como por la interacción entre los mismos estudiantes. Pero quizás la mayor ventaja que proporciona la enseñanza online es la individualización de los contenidos, esto rompe el clásico esquema de las clases, donde todos los alumnos deben seguir el ritmo impartido por el profesor lo cual es un problema para algunos alumnos con problemas en la asignatura y para otros con más conocimientos sobre la asignatura puede ser una causa para la perdida de interés en la misma.

La individualización consiste en permitir a los alumnos escoger ellos mismos el ritmo al que desean obtener los nuevos conocimientos. Esto es mucho más sencillo de conseguir en una aplicación que sigue el progreso individual de cada alumno y pone a disposición del mismo el temario dado en clase, junto con actividades para reforzarlos.

La motivación de este proyecto es la inclusión de un módulo en una aplicación de aprendizaje online, que permitirá realizar el proceso de inclusión de contenido así como de edición de los mismos, de una manera rápida y eficaz colaborando así a crear un proceso educativo sencillo y usable por todas las personas que desean aprender, siendo esta en especial, una oportunidad para impulsar el desarrollo de este tipos de tecnologías dentro del entorno de la Universidad Autónoma de Madrid.

1.2 Objetivos

El objetivo de este Trabajo de Fin de Grado es la implementación (diseño y desarrollo) de un nuevo módulo para una aplicación de introducción a la programación en el lenguaje C.

El objetivo del proyecto es disminuir significativamente el tiempo que se emplea para añadir nuevo temario a la aplicación y reducirá la probabilidad de aparición de errores en este proceso, ya que actualmente esta tarea se ha de realizar de forma manual editando directamente el código fuente de la aplicación. Este mismo módulo permitirá además la edición de las unidades que ya están presentes en la aplicación.

Para la consecución del objetivo principal se pueden distinguir una serie de hitos explicados esquemáticamente a continuación:

- Dado que se requiere modificar el código existente de la aplicación, el primer paso es comprender el funcionamiento de la aplicación actual y las decisiones de diseño que se tomaron.

- Una vez comprendido el funcionamiento, es necesario realizar un diseño preliminar del nuevo módulo y comprobar si es inevitable realizar algún cambio en el actual código.

- Se realizara un estudio sobre la profundidad de los cambios en los módulos ya desarrollados.

- Se comenzará el desarrollo del nuevo módulo de creación de unidades de manera independiente a la aplicación actual, para que esta pueda continuar con su normal funcionamiento.

- Una vez que el módulo sea totalmente funcional se añadirá la funcionalidad que permite editar las unidades, este paso es sencillo ya que al tratarse de un modelo vista controlador, es posible reutilizar el código y usar la misma estructura variando simplemente los datos.

- Se integrará el nuevo módulo con la aplicación completa.

- Se realizaran pruebas para comprobar el correcto funcionamiento global de la aplicación.

1.3 Organización de la memoria

Este documento está dividido en apartados que se corresponden con las distintas fases del desarrollo del software.
Comienza con una breve descripción de las tecnologías usadas en el proyecto así como el estado de software similar y como se encuentra estructurado (fase de análisis).

A continuación se encuentran los capítulos dedicados a las fases de diseño, tanto el modelo de ciclo de vida escogido para su desarrollo como cada uno de los diseños de sus componentes (código, bases de datos…).

El capítulo de desarrollo está centrado en el control y manejo de los datos que el usuario introduce en la aplicación y como se gestionan errores.

Por último se encuentran los capítulos de seguridad, errores, integración y pruebas, donde se detallan los pasos finales del proyecto para su completa incorporación a la aplicación.

2 Estado del arte

Antes de comenzar el desarrollo del nuevo módulo de la aplicación, es necesario estudiar las distintas plataformas similares que actualmente se encuentran en funcionamiento así como la tecnología que usan, tanto dichas aplicaciones, como sobre la que se realizara todo el trabajo, de esta manera es posible seguir patrones de diseño comunes al desarrollo de este tipo de plataformas.

2.1 Análisis sobre las plataformas de aprendizaje online

Una plataforma de aprendizaje online es una aplicación software (en nuestro caso nos centraremos en las aplicaciones web) que permite automatizar la administración y seguimiento de los contenidos didácticos. (2)
Generalmente todas estas plataformas ofrecen unas prestaciones similares (3) tales como:

- Centralizar y automatizar la administración de contenidos (así como permitir la personalización de los mismos).
- Gestión de informes.
- Entrega y/o corrección de ejercicios.
- Parte social, que puede y debe incluir alguna herramienta de discusión sobre los contenidos.
- Sistema de puntuación.

Algunas de estas plataformas online como Lynda o EDX ofrecen cursos interactivos y personalizados en los que cada alumno puede escoger el tiempo en el que desea terminarlos, poniendo a su disposición todo el temario. Otra de las claves de este sistema es que no importan los estudios que tengas, siempre puedes acceder a aéreas de conocimiento distintas a las tuyas, de tal manera que un ingeniero puede adquirir conocimientos sobre economía aumentado la libertad del estudiante y no haciéndole sentir "encajonado" en su área.

Como hemos podido comprobar una de las funcionalidades más importantes de una aplicación de aprendizaje online es la gestión de los contenidos que son accedidos por los estudiantes. En el caso de este proyecto, esta es la parte que se ha desarrollado en su totalidad.

2.2 Análisis de las Tecnologías utilizadas

Actualmente existen numerosos lenguajes de programación y tecnologías usadas en el desarrollo de aplicaciones web, algunos de estos lenguajes son usados tanto para programación de aplicaciones de escritorio como desarrollo web.

Si estudiamos las estadísticas de uso sobre los lenguajes de programación, podemos observar claramente que el liderazgo lo tienen los lenguajes que permiten su uso tanto para el desarrollo de aplicaciones de lado del servidor, como aplicaciones móviles o aplicaciones para PC, los lenguajes líderes usados en este contexto son Java, C#, .NET.

Nov 2016	Nov 2015	Change	Programming Language	Ratings	Change
1	1		Java	18.755%	-1.65%
2	2		C	9.203%	-7.94%
3	3		C++	5.415%	-0.78%
4	4		C#	3.659%	-0.66%
5	5		Python	3.567%	-0.20%
6	8	▲	Visual Basic .NET	3.167%	+0.94%
7	6	▼	PHP	3.125%	-0.12%
8	7	▼	JavaScript	2.705%	+0.23%
9	11	▲	Assembly language	2.441%	+0.56%

Ilustración 1 Lenguajes de programación más usados (Nov 16). Fuente: tiobe.com (4).

No obstante, es muy notable encontrar lenguajes específicos (ya sea para realizar aplicaciones de escritorio o para desarrollo web) que tienen un alto porcentaje de uso. Estos lenguajes al tener un propósito mucho más definido permiten una mayor "potencia".
Una de las fases previas al desarrollo de los nuevos módulos es la compresión del diseño y funcionamiento de la aplicación actual y para ello se ha realizado un estudio sobre las tecnologías utilizadas en la misma, siendo las más notables PHP, en concreto mediante el uso del framework Laravel, JavaScript, SQL y HTML.

2.2.1 HTML

HTML es el lenguaje de marcas estándar para la creación de páginas web. Forma junto con JavaScript y CSS una de las tecnologías en las que se basa actualmente todo el contenido que se distribuye a través de la World Wide Web. Estas marcas (comúnmente llamadas tags HTML) son fácilmente distinguibles del contenido y su objetivo es la descripción de la estructura de la página web mediante bloques.

El comienzo de un bloque en HTML se denota mediante un tag de apertura (<head>,) y un tag de cierre que generalmente se escribe igual que el tag de apertura con una barra inclinada precediendo al nombre del tag (</head>,), entre ambos tags (apertura y cierre), se encuentra el contenido de la página, que a su vez puede contener más tags anidados.

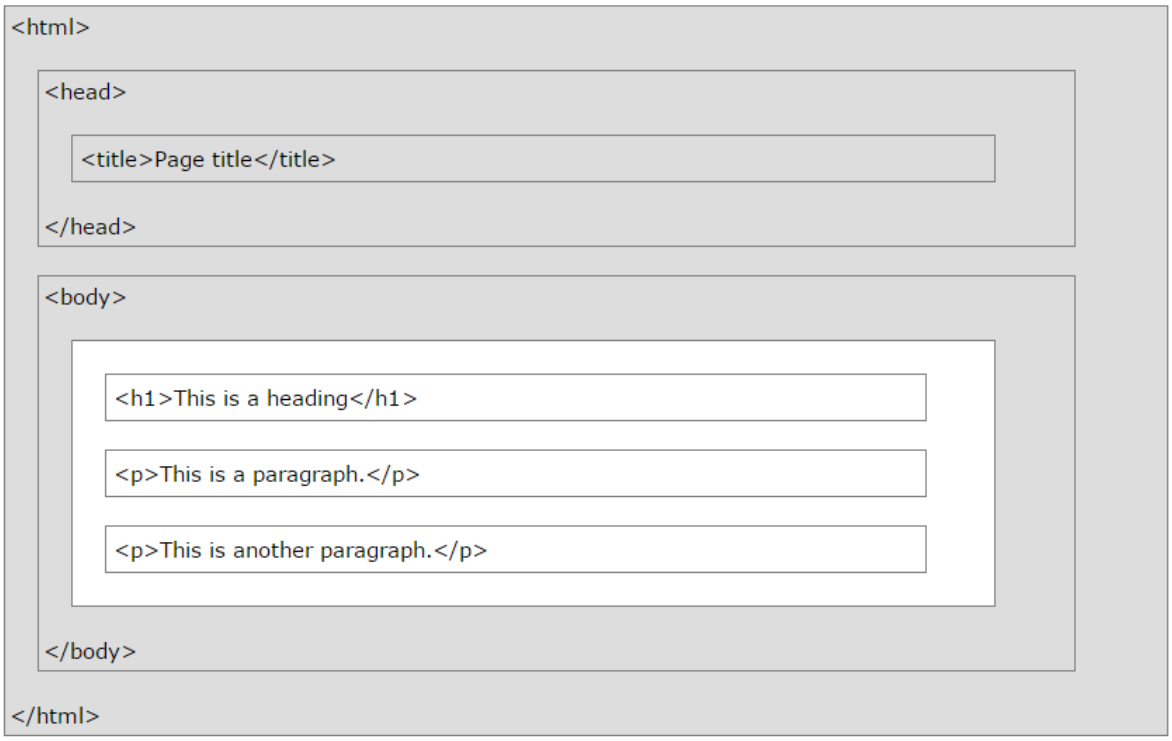

Ilustración 2 Ejemplo de la estructura de una página HTML.

2.2.2 JavaScript

JavaScript, es un lenguaje interpretado de programación de alto nivel. Está basado en una especificación estándar del motor del lenguaje llamada ECMAScript (5), en la cual se basan todas sus actuales variantes (JScript, ActionScript, JavaScript).

Hoy en día la mayoría de los navegadores de internet implementan la variante llamada JavaScript incluyendo un intérprete del mismo en el propio navegador, esta modificación de la especificación ECMA l incluye importantes añadidos no incluidos en la especificación original.

Aunque JavaScript también es usado fuera del entorno del desarrollo de aplicaciones web, su uso primario es en aplicaciones de internet y más en concreto, del lado del cliente.

JavaScript otorga versatilidad a las páginas web, ya que permite la modificación de elementos del DOM (HTML) una vez que la página ha sido cargada en el cliente, permitiendo evitar nuevas y costosas peticiones al servidor.

No obstante presenta una gran desventaja con respecto a la seguridad, ya que al tratarse de código ejecutado en el lado del cliente, este código es vulnerable a modificaciones y por

tanto cualquier dato que se quiera almacenar en la aplicación debe ser redundantemente comprobado tanto por JavaScript (lado del cliente) como por lado del servidor.

2.2.3 PHP

PHP, es un extendido lenguaje de programación interpretado (no compilado) de código abierto, cuyo propósito general es el desarrollo de aplicaciones web (6). El código en PHP se ejecuta del lado del servidor, de tal manera que cuando un cliente realiza una petición de una página web el modulo del intérprete PHP genera un código HTML, a partir de los ficheros PHP, que puede ser visualizado como una página web en cualquier navegador.
Esta versatilidad a la hora de generar código HTML permite realizar aplicaciones web con contenido dinámico.

A partir de la versión 3.0 se introdujo en PHP el soporte para la programación orientada a objetos, lo cual otorga aún más versatilidad a la hora de desarrollar aplicaciones web. Actualmente se encuentra en la versión 7.0 y aunque recibe actualizaciones usualmente, su uso se está viendo reducido debido a la gran limitación de PHP, se trata de un lenguaje basado en peticiones y esto limita la interactividad del usuario con la página web.

2.2.4 Laravel

Laravel es un framework web, basado en PHP, el cual está destinado a facilitar el desarrollo de aplicaciones web siguiendo el "Modelo Vista Controlador" (MVC).

Algunas de las características en las que se basa Laravel son (7):

- Mejorar el acceso a las bases de datos, creando un sistema más directo, tanto para realizar consultas como para acceder a los datos devueltos al ejecutar las mismas.
- Separar la interfaz gráfica de la lógica y datos de la aplicación mediante vistas y controladores.
- Aprovechar las características de programación orientada a objetos que permite PHP.
- Posibilidad de crear plantillas a partir de modelos de datos.
- Generación de pruebas unitarias para realizar tests sobre la aplicación.

2.2.5 SQL

SQL es un lenguaje de programación diseñado para mantener los datos de un sistema de bases de datos relacionales y que permite especificar operaciones con ellos (inserción, actualización). Enfocado a la aplicación sobre la que trata este Trabajo de Fin de Grado, SQL permite el acceso mediante consultas (Queries) a los datos almacenados en la Base de datos, así como la modificación, eliminación o inserción de los mismos (8).

3 Diseño

Este apartado comienza con una explicación sobre el modelo de ciclo de vida del software escogido para este Trabajo de Fin de Grado. A partir de él se exponen los detalles de cada una de las fases de Análisis de requerimientos, objetivos y diseño general. Ya que se trata de una aplicación ya desarrollada es importante recalcar que muchas de las decisiones de diseño están tomadas a partir del antiguo diseño de esta aplicación, aunque se incluido algunos cambios sustanciales que serán explicados en este apartado.

3.1 Modelos de ciclo de vida del software

Antes de comenzar con cada una de las fases típicas del desarrollo de una aplicación, es necesario escoger adecuadamente un modelo para el desarrollo. Dado que existen numerosos modelos distintos, pero en concreto existen tres que encajan adecuadamente con los requisitos de desarrollo del nuevo módulo de la aplicación. Creo que es apropiado nombrarlos brevemente con una pequeña explicación sobre porqué encajan con el proyecto y finalmente se incluye cual ha sido el modelo escogido junto con una justificación explicando porque es el más acertado.

Podemos distinguir tres enfoques distintos que son adecuados a los requisitos del desarrollo del nuevo módulo de la aplicación (9):

- Desarrollo en cascada.
- Desarrollo en espiral.
- Desarrollo incremental.

El **desarrollo en cascada** consiste en realizar un desarrollo secuencial de las distintas fases del proyecto. Este enfoque no está indicado para un proyecto de tipo incremental, es decir una vez terminada cada una de las fases fase se desaconseja volver a una anterior, en el caso de que quisiéramos aplicarlo a nuestro proyecto, ya que actualmente nos encontramos en la fase de Mantenimiento, el enfoque a tomar sería el de añadir nuevas fases de diseño, implementación y pruebas, lo cual no está contemplado en una aplicación estricta de esta metodología.

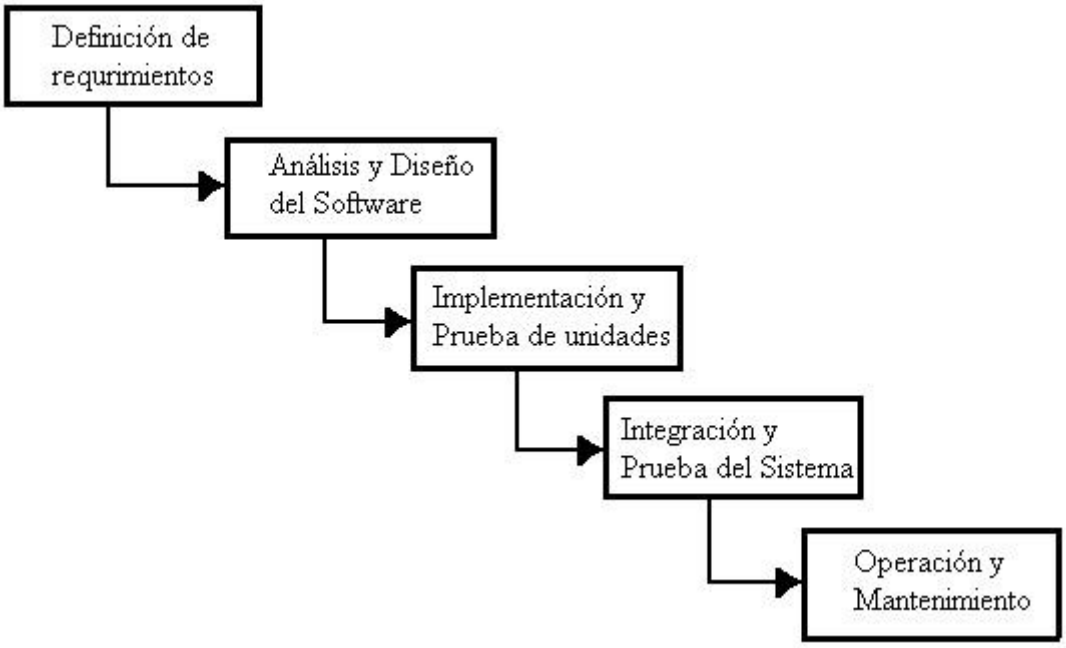

Ilustración 3 Fases del desarrollo en cascada Fuente: http://isescom.blogspot.com.es/

El **desarrollo en espiral** incluye aspectos claves del desarrollo en cascada, consiste en la aplicación de una serie de fases similares a las del desarrollo en cascada que son aplicadas en ciclos, antes de finalizar cada ciclo, se planea la siguiente iteración, esto permite reevaluar los riesgos y requisitos de una aplicación y modificarlos de manera que se ajusten a la realidad y al ritmo de desarrollo del proyecto (10).

El **desarrollo incremental** basa su eficacia en dividir un proyecto grande en segmentos más pequeños, esto reduce los riesgos del proyecto, ya que al crear módulos independientes, la aplicación podría ser totalmente funcional sin haber desarrollado todos los módulos que la forman. Como el modelo en espiral, este método también se basa en la idea de desarrollar un sistema mediante la repetición de fases (11), la principal diferencia con el modelo en espiral es que cada una de estas fases es independiente de la anterior (recordemos que en el desarrollo en espiral estas iteraciones son reajustadas antes de comenzar la siguiente).

3.1.1 Modelo de ciclo de vida del software escogido

El diseño del nuevo módulo de la aplicación se puede dividir en dos grandes fases. Ya que es necesario trabajar sobre software ya desarrollado el cual siguió su propia metodología de diseño, el primer paso es estudiar cuál de todos los patrones de diseño se optó por seguir y cuáles fueron las razones por la que se tomó tal decisión, la segunda fase consiste en el desarrollo e integración de este nuevo módulo con la aplicación completa.
Tras una fase de estudio preliminar de la aplicación actual, para la cual se ha examinado cada uno de los módulos de la aplicación así como la interconexión entre los mismos se han recabado esquemáticamente los siguientes datos sobre el tipo de metodología aplicada en su desarrollo:

- La aplicación está dividida en módulos.
- Cada uno de estos módulos es independiente.
- Se sigue un patrón MVC.

Es fácil advertir que se optó por un proceso de desarrollo de software **incremental**, lo cual facilita la labor a la hora de añadir nueva funcionalidad a la aplicación.
Para el desarrollo de este proyecto se ha optado por continuar con este modelo de ciclo de vida del software, teniendo como objetivo que el desarrollo de este módulo se complete como una iteración del mismo. Las fases a aplicar se enumeran a continuación y cada una de ellas será explicada a lo largo de los siguientes apartados:

- Fase de Análisis de requisitos.
- Diseño
- Implementación
- Integración
- Pruebas

Como se puede comprobar algunas de estas fases son compartidas por los otros modelos de ciclo de vida del software, la principal diferencia radica en que al tratarse de módulos independientes toma un mayor peso la fase de integración.

3.2 Análisis de Requisitos

En este apartado se listan, acompañados de una breve descripción, cada uno de los requisitos que debe cumplir el nuevo módulo de la aplicación, para ello se han diferenciado los requisitos funcionales, que identifican las mínimas funciones a realizar por el módulo para que este pueda ser incluido en la aplicación, y los requisitos no funcionales, que están relacionados con aspectos secundarios que sería muy recomendable incluir en el proyecto, pero no necesarios para su funcionamiento.

Requisitos Funcionales

- Debe permitir añadir nuevo contenido en la aplicación a través de un formulario. Este requisito puede ser dividido en los siguientes:
    - El contenido ha de estar dividido en Unidades y dentro de cada unidad pueden existir un número de páginas variable.
    - Cada página ha de incluir un título, un contenido formado por texto y/o imágenes y una sección de código de un programa (en C).
    - El código del programa se ha de cargar de tal manera que permita su compilación y ejecución mediante el módulo ya desarrollado en la aplicación.
    - El código a mostrar en cada página debe poder modificarse por el usuario.
    - Cada página puede incluir preguntas, estas se almacenaran junto con las respuestas, así como cuál de todas ellas es la correcta.
- Debe poderse editar contenido ya almacenado en la aplicación.
- Solo los usuarios con los permisos adecuados pueden editar o añadir nuevo contenido.

Requisitos no funcionales

- Existen textos de ayuda que deben ser guardados junto con el resto de contenidos y mostrados al usuario cuando sea necesario.
- En las preguntas, el orden de las respuestas debe mostrarse aleatoriamente.
- Cada página puede incluir múltiples preguntas.
- La interfaz gráfica para introducir los nuevos datos debe ser intuitiva para el usuario final.

3.3 Análisis de los módulos funcionales de la aplicación

Como se explicaba anteriormente, la primera de las grandes fases de este proyecto consiste en analizar los módulos ya funcionales antes del desarrollo de este Trabajo de Fin de grado. Estos módulos (siguiendo el modelo de desarrollo incremental) son casi independientes entre sí, pero algunos dependen directamente del nuevo módulo de gestión contenidos de la aplicación que se ha realizado y fue necesario rediseñarlos, en concreto el módulo de visualización de unidades tuvo que ser diseñado y desarrollado por completo.

3.3.1 Cambios en el sistema de visualización de unidades

En esta sección se estudiaran los cambios en el diseño que han sido necesarios en el módulo de visualización de unidades de la aplicación.
Para la visualización del temario almacenado en la aplicación, y más en concreto para solicitar los datos de una unidad, es necesario realizar una petición a la url "/unidad{id}" siendo **id** un parámetro obligatorio de tipo entero, que indica el número de unidad que se ha de cargar.

El controlador asignado a esta ruta realiza una pequeña carga de datos sobre el progreso del usuario y carga un fichero JavaScript diferente por cada unidad, también devuelve una vista distinta por cada una de las unidades.

Dentro de los ficheros JS se almacenan los siguientes datos de cada una de las páginas de la unidad:

- El código (en lenguaje C) que el usuario puede compilar y ejecutar. Este código está almacenado como un array de strings.
- Los títulos de cada una de las páginas, en concreto los títulos de cada uno de los textos.
- Los textos de ayuda, también almacenados como un array de Strings.
- También se incluyen otros datos generales como que páginas incluyen preguntas, el número total de páginas que forman la unidad…

```
var codearray = [
    "#include <stdio.h>\nint main(void)\n{\n\tprintf(\"Hola\");\n\n\treturn 0;\n}",
    "#include <stdio.h>\nint main(void){printf(\"Hola\"); return 0;}",
    "#include <stdio.h>\nint main(void)\n{\n\tprintf(\"Hola\");\n\n\treturn 0;\n}",
    "#include <stdio.h>\nint main(void)\n{\n\tprintf(\"Hola\");\n\n\treturn 0;\n}",
    "#include <stdio.h>\nint main(void)\n{\n\tprintf(\"Hola\");\n\n\treturn 0;\n}",
    "#include <stdio.h>\nint main(void)\n{\n\tprintf(\"Hola\");\n\n\treturn 0;\n}",
    "#include <stdio.h>\nint main(void)\n{\n\tprintf(\"Hola\");\n\n\treturn 0;\n}",
    "#include <stdio.h>\nint main(void)\n{\n\tprintf(\"Hola\");\n\n\treturn 0;\n}",
    "/*\n * Fichero: main.c\n * Autor:   Alejandro Sierra\n *\n * Creado el 22 de mayo de 2016, 18:54\n
    */\n\n#include <stdio.h>\nint main(void)\n{\n\tprintf(\"Hola\");\n\n\treturn 0;\n}",
    "/*\n * Fichero: main.c\n * Autor:   Alejandro Sierra\n *\n * Creado el 22 de mayo de 2016, 18:54\n
    */\n\n#include <stdio.h>\nint main(void)\n{\n\tprintf(\"Hola\");\n\tprintf (\"¿Cómo estás?\");\n\n\treturn
    0;\n}",
    "","",""];

var titles=["Buenas tardes","El estilo es importante","No puede faltar main","La cabecera de main","El cuerpo de
main","El compilador",
    "El enlazador","Avisos y errores","Comentarios","Salto de línea","El estándar","Ayuda"];

var helps=["",
        "El estilo correcto es el siguiente:</br></br>#include &ltstdio.h&gt</br>int main (void)</br>{
        </br>  printf (\"Hola\");</br>  return 0;</br>}",
        "Sustituye el nombre main por area, sin acento. A continuación, ejecuta el programa. Como podrás observar,
        surge un error. Este error es debido a que todo programa en C necesita una función de nombre main.",
        "Como ves, la aplicación te devuelve un mensaje indicando que falta especificar el tipo de dato que
        devuelve main. Este tipo debe ser siempre int.",
        "Debes sustituir el texto Hola por la frase tal y como quieres que se muestre, incluidos los espacios en
        blanco entre palabras.",
        "","","",
        "Para comentar una sola línea, lo más económico es colocar el símbolo // al comienzo.",
        "Coloca los símbolos \\n justo al final de la cadena del primer printf del programa."
    ];
var totalpages=12;
var executionpages=[5,10];
var compilepages=[];
var testpages=[1,3,4,6,7,8,9,11];
var checkexec=[5,10];
var checkcode=[];
var unitid=1;
```

Ilustración 4 Ejemplo de unos de los Ficheros JavaScript.

Dentro de los ficheros PHP que forman las vistas, se encuentran estos datos:

- Texto de cada una de las páginas.
- Preguntas y respuestas de los ejercicios.

En forma esquemática, este proceso se resume de la manera siguiente.

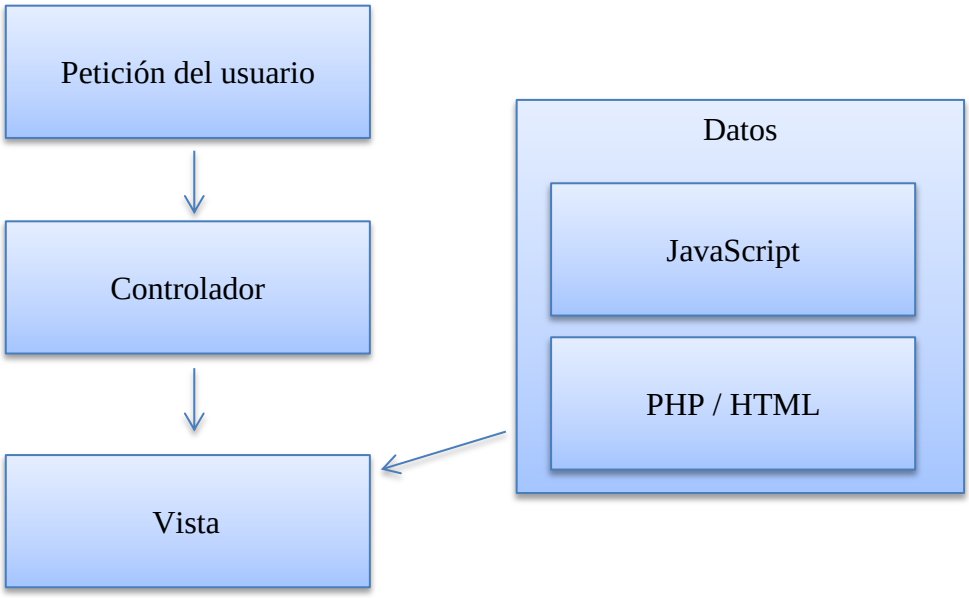

Ilustración 5 Flujo de los datos tras una solicitud del usuario.

Como se puede observar los datos están repartidos entre dos fuentes distintas. Así mismo es fácil comprobar que es necesario crear ficheros JavaScript y PHP por cada una de las unidades, esto significa que hay que respetar una sintaxis y formato concreto, lo cual rompe el esquema MVC que dicta que los datos deben de ser independientes de la interfaz encargada de las vistas y que es la que presenta los datos finales al usuario.

Es por ello que se ha optado por rediseñar completamente este módulo para aproximarse más a un diseño MVC, para ello, los datos serán desplazados a la Base de Datos, desde donde serán cargados directamente, de este modo todos los datos siguen la misma estructura, esto a su vez otorga una ventaja sobre el desarrollo del módulo de edición de unidades, ya que ahora este módulo solamente debe modificar los datos dentro de estas tablas de la Base de datos. En cambio si se hubiera optado por seguir el antiguo diseño el módulo de edición de unidades tendría que generar ficheros JavaScript y PHP, tarea nada sencilla y mucho menos a la hora de editarlos.

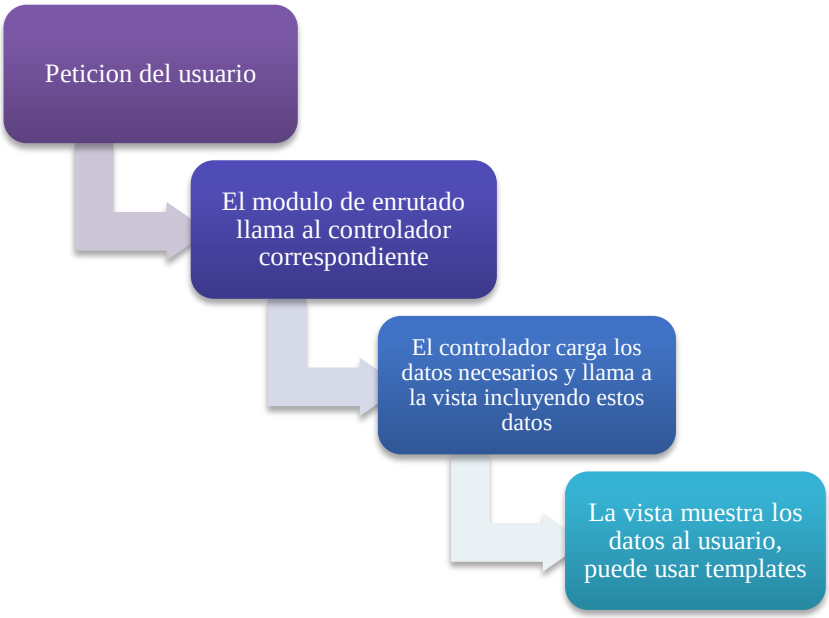

Ilustración 6 Fases de carga de datos tras una peticion.

3.4 Diseño de la Base de Datos

En este apartado se exponen los cambios que han sido necesarios realizar en la base de datos. Estos cambios han sido necesarios para agrupar todos los datos bajo una misma fuente.

3.4.1 Antiguo Diseño

El antiguo diseño de la base de datos constaba de cinco tablas. Muchas de estas tablas se han mantenido iguales casi en su totalidad en el nuevo diseño, por ello, a continuación se explica brevemente la funcionalidad de cada una:

- **Users**: Almacena todos los datos relacionados con los usuarios y su información de acceso.

- **Progress**: En esta tabla se encuentra la información correspondiente al progreso de cada uno de los usuarios, tiene una relación 1 a 1 con la tabla **Users,** mediante la id del usuario.

- **Bugs**: Cada vez que un usuario encuentra un fallo en la aplicación, puede reportarlo mediante el botón correspondiente, la información de cada uno de los reportes se almacena en esta tabla, tiene una relación 1 (usuario) a muchos (opcional) (bugs) mediante la id del usuario.

- **Statistics**: almacena las estadísticas de los usuarios tales como el tiempo empleado en resolver ejercicios o el número de intentos, tiene una relación 1 (usuario) a muchos (opcional) (bugs) mediante la id del usuario.

- **Solutions:** En ella se encuentran las soluciones de los ejercicios. No posee relación con las demás tablas. Hay que destacar que esta tabla solo almacena la respuesta correcta al ejercicio de cada página, el resto de respuestas, junto con los otros datos de las unidades se almacenan en ficheros (JavaScript, PHP/HTML) tal y como se comentó en la sección anterior.

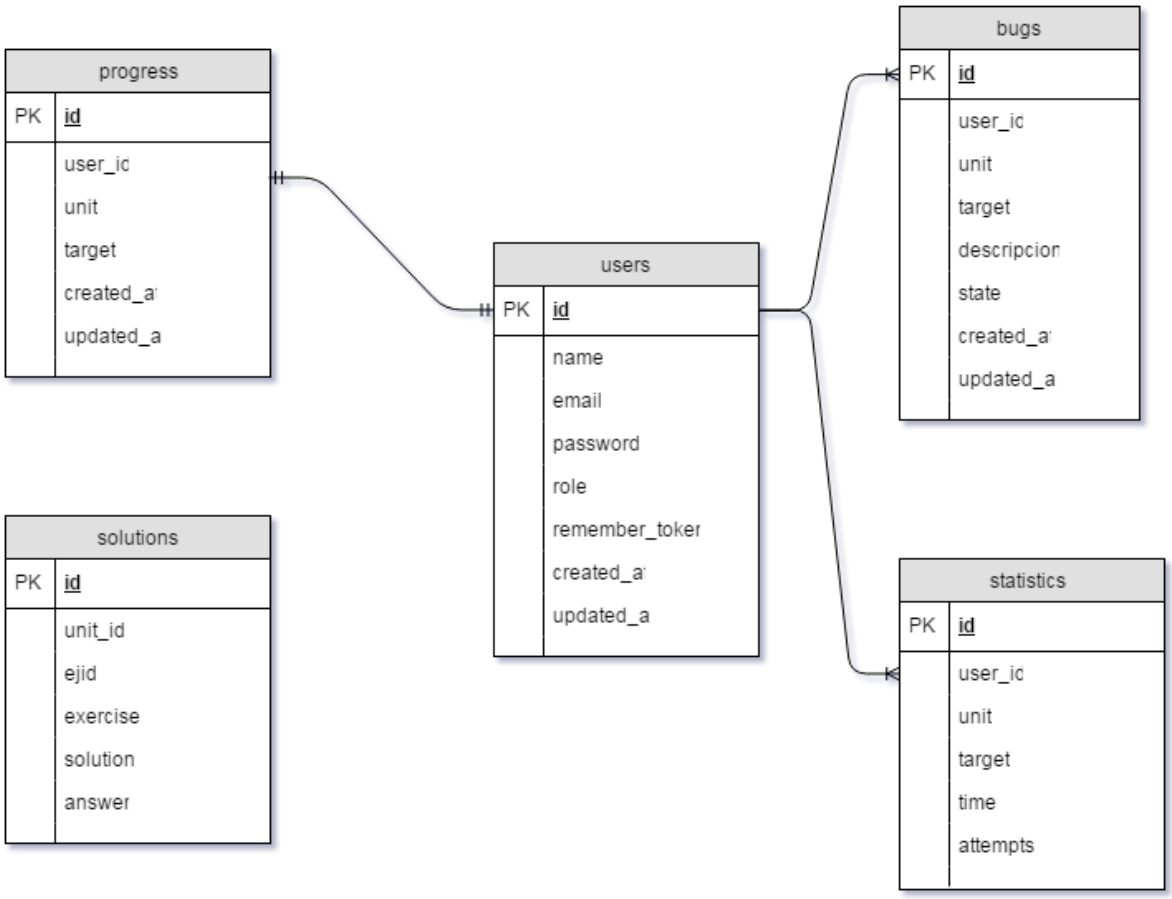

Ilustración 7 Diagrama ER de las antiguas tablas base de Datos.

3.4.2 Nuevo Diseño

Siguiendo el modelo de diseño escogido para este proyecto, es evidente que es necesario mover los datos de ficheros que no siguen un formato concreto a algún tipo de almacenamiento que nos proporcione un modelo de datos "estándar" y homogéneo. Una primera aproximación podría ser crear un módulo que generase estos ficheros almacenando los datos que queremos en un formato definido por unas sencillas reglas (CSV). Pero esto es ineficiente y no nos proporciona relaciones entre los datos.
Por ello, lo más adecuado es modificar la base de datos existente para acomodarla a los datos que queremos almacenar, esto representa dos grandes cambios con respecto al diseño expuesto en el apartado 3.4.1(página 15).

- Se han de crear nuevas tablas. Esto es necesario porque es imposible almacenar los datos correspondientes a unidades, páginas y ejercicios en la actual base de datos.

- La tabla **solutions** está obsoleta. Recordemos que esta tabla **solo** almacenaba las respuestas a los ejercicios, mientras que el resto de respuestas está en los ficheros. Esto incumple las reglas básicas del modelo MVC, ya que los datos están divididos entre múltiples fuentes.

Las dos nuevas tablas añadidas son las siguientes

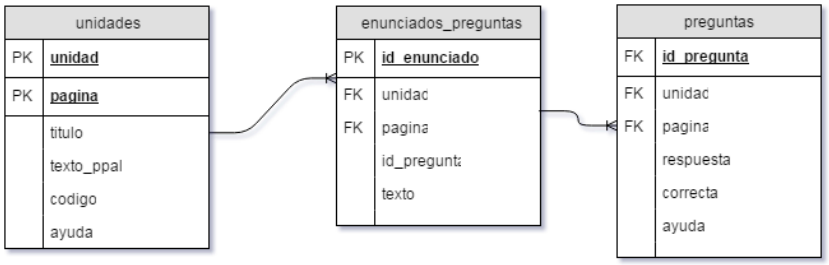

Ilustración 8 Tablas añadidas a la base de datos.

- **Unidades**: Almacena un registro por cada una de las páginas de contenido, tiene una clave primaria compuesta por unidad + página. Esta tabla unifica los datos que antiguamente estaban divididos entre los ficheros JavaScript y PHP, de tal manera que ahora solo es necesario acceder a una sola fuente de datos (una tabla en la DB) para acceder a los mismos.

- **Enunciados_preguntas**: Esta tabla almacena todos los enunciados de las preguntas de cada una de las páginas. Dado que es posible que haya varias preguntas por página se ha creado la columna id_pregunta, que enlaza con la tabla de respuestas.

- **Preguntas**: Contiene todas las respuestas a los ejercicios de cada una de las páginas (en el caso de que la página tenga un ejercicio disponible). En el caso de que una fila de esta tabla contenga los datos de una respuesta correcta simplemente contendrá en el campo correcta = TRUE. Esta tabla unifica la antigua tabla solutions, que solo contenía las respuestas correctas a las preguntas, ahora tanto las respuestas correctas como las incorrectas se encuentran almacenadas en el mismo lugar.

La nueva base de datos quedaría conformada de la siguiente manera:

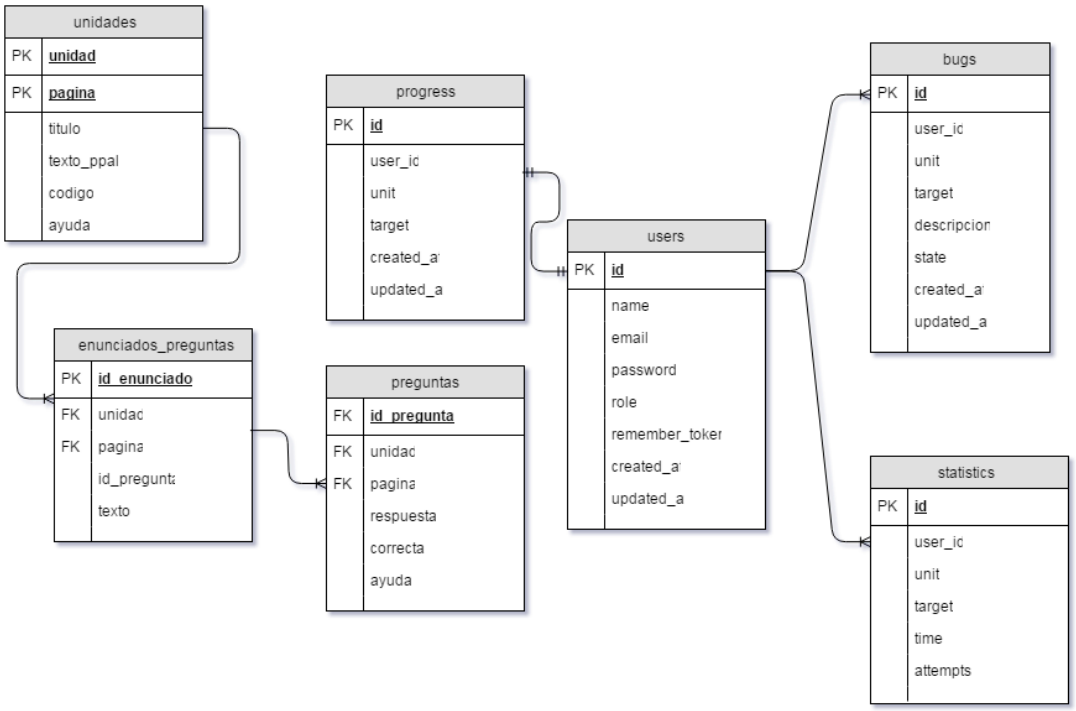

Ilustración 9 Diagrama ER de la base de datos completa.

3.5 Diseño de la interfaz

Este proyecto trata de facilitar la entrada de contenidos a la aplicación por parte de los usuarios, la forma de que el usuario pueda introducir estos datos fácilmente es mediante una interfaz gráfica a través de un formulario web. Es por ello por lo que esta parte de la aplicación es de gran importancia.

3.5.1 Diseño preliminar de la interfaz

En la primera fase de diseño se realizó un esquema preliminar del formulario, en el existía una pantalla principal donde se solicitaba al usuario los datos sobre el tema que deseaba añadir o editar, además se solicitaba al usuario el número de páginas (y ejercicios) que esa unidad tendría.
Dentro de la unidad se generarían páginas (o ejercicios) en blanco, según el número solicitado por el usuario, y finalmente el usuario las rellenaría una por una.

Al final de este proceso el usuario habría introducido los siguientes datos:

Unidad

- Titulo Unidad.
- Número de páginas que la forman.
- Texto de introducción.

Pagina

- Texto de introducción.
- Ejercicio.
- Preguntas
- Código
- Texto de ayuda.

Ilustración 10 Diseño preliminar del formulario de edición de contenidos.

Como se puede observar en la figura este diseño presenta una deficiencia importante. Si nos fijamos en la lista de datos que el usuario debe introducir, podemos observar que a la hora de introducir los datos de la Unidad se solicita al usuario el número de páginas que la conforman.
A pesar de que esta restricción no incumple con ninguno de los requisitos funcionales de la aplicación, conocer de antemano el número de páginas que ocupará un tema concreto quizás no sea posible.
Se han hecho modificaciones en la interfaz, para solucionar esta y otras deficiencias así como para hacerla más sencilla de usar por parte del usuario final.

3.5.2 Diseño final de la interfaz

Las modificaciones con respecto al anterior diseño radican en que ahora las páginas se van añadiendo dinámicamente tras completarlas, de esta manera el usuario puede parar en cualquier momento y todas las paginas añadidas hasta ese instante estarán ya guardadas en la aplicación. Este sistema también presenta la ventaja de que no es necesario completar una unidad entera de una vez, sino que es posible irlas alternando e ir añadiendo páginas más adelante.

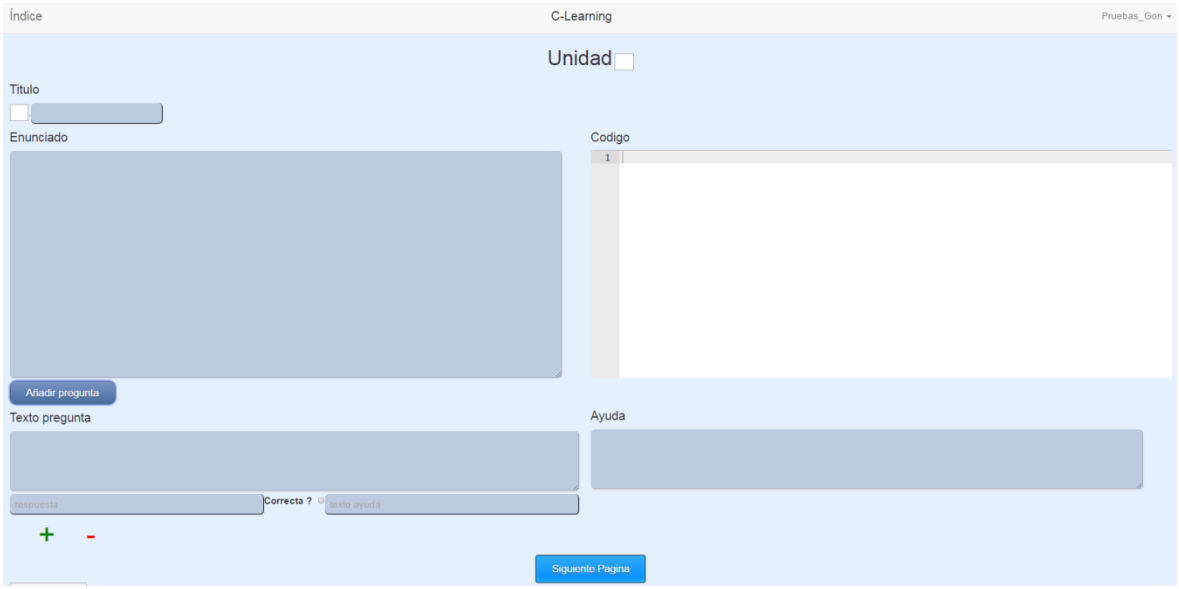

Ilustración 11 Diseño final de la interfaz del formulario.

3.6 Diagrama de clases

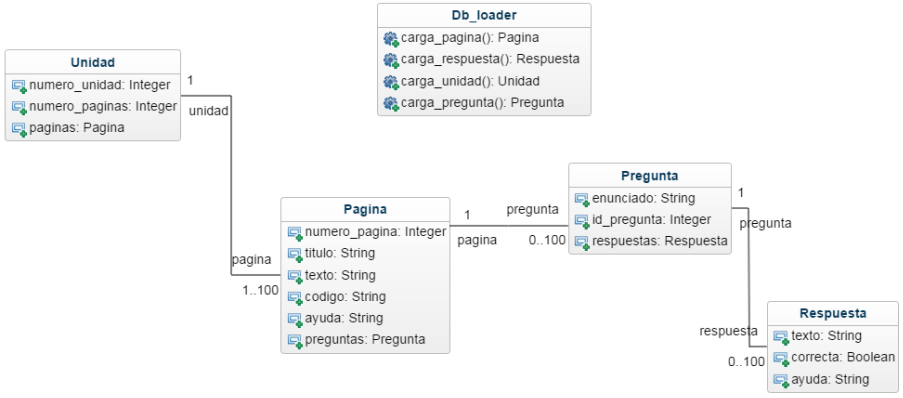

Ilustración 12 Diagrama de clases de la aplicación.

Con el propósito de utilizar la funcionalidad orientada a objetos de PHP se ha realizado un diseño de cada una de las clases que componen el proyecto así como sus relaciones entre las mismas, a continuación se explica paso a paso cada una de ellas así como las razones por las que se siguió este diseño.

Las clases Unidad, Página, Pregunta y Respuesta almacenan la información correspondiente de la base de datos. Como se aprecia en el diagrama, cada una de las clases contiene una variable de tipo Array de elementos de la clase relacionada con ella, es decir, la clase Unidad tiene una variable (paginas) de tipo Array siendo cada uno de los elementos de este Array de tipo Página. Esta misma lógica se repite de la misma manera para la clase Página, que contiene un Array de Pregunta, y para la clase Pregunta, que contiene un Array de Respuesta. Todas estas relaciones son una(o ninguna) a muchas, ya que una unidad puede contener cualquier número de páginas, así mismo una Página puede contener cualquier número de Preguntas y por último cada pregunta o ejercicio tiene cero o más respuestas (en el caso de los ejercicios sin respuestas, son simplemente ejercicios que se realizan sobre el código de la propia página).

La Clase Db_loader, en cambio, es totalmente diferente a las demás, esta sirve de intermediario entre la base de datos y el modelo de datos de la aplicación. Esta clase incluye todos los métodos que permiten cargar cada una de las filas de la BD y transformarlas a Objetos que pueden ser usados en la aplicación (Clases).

Existe una clase adicional para la gestión de errores que será explicada en el apartado de control y manejo de errores.

4 Desarrollo

Durante la fase de desarrollo del proyecto se llevó a cabo la programación del nuevo módulo cumpliendo los requisitos funcionales recogidos en la fase de diseño. Ya que la estructura de este proyecto fue explicada anteriormente este apartado en concreto hace hincapié en el desarrollo de las partes de los módulos que gestionan la entrada de datos por parte del usuario y el control de los posibles errores.

4.1 Modificaciones del módulo de visualización de unidades

Tras rediseñar la manera en la que se almacenan los datos, fue necesario como ya se ha explicado realizar un nuevo módulo que permitiese acceder a estos datos y mostrarlos de la misma manera que se hacía anteriormente.
Algunos de los componentes de este módulo de visualización se han podido mantener, pero otros ha sido necesario desarrollarlos de nuevo por motivos de incompatibilidad no subsanables.

Control de soluciones a los ejercicios
Como se explicaba en el apartado 3.4.2, la tabla que contiene las soluciones a los ejercicios está ahora obsoleta y por tanto también lo está la forma mediante la que se comprueba si un ejercicio está resuelto correctamente o no.
Para actualizar esta funcionalidad se ha modificado el controlador asociado para que obtenga los datos de la tabla adecuada de la base de datos, guarde las estadísticas correspondientes al número de intentos en la BD (tabla de estadísticas) y envíe de nuevo al cliente un mensaje indicando si la respuesta seleccionada es la correcta o es incorrecta, junto con un texto de ayuda.

Control de textos de ayuda de la unidad
Cuando se muestra una unidad, el alumno puede solicitar ayuda sobre el temario correspondiente pinchando el botón adecuado. Esta información también ha sido desplazada por lo que la manera de mostrarla al usuario también ha cambiado. Ahora se usa un controlador intermedio dado que los textos de ayuda ya no se encuentran almacenados en los ficheros JavaScript, sino en la base de datos. Este controlador procesa estos datos y los envía al cliente para que puedan ser mostrados.

4.2 Control y manejo de errores

En este proyecto y ya que el modulo en concreto se basa en la entrada de datos, existen dos partes diferenciadas donde pueden producirse y gestionar los errores, en el lado del cliente y errores en el lado del servidor.
Tras producirse un error antes de enviar datos al servidor, es posible mostrar datos sobre el mismo por pantalla para que el usuario pueda corregirlo, en cambio los errores generados del lado del servidor han de ser gestionados de manera diferente.

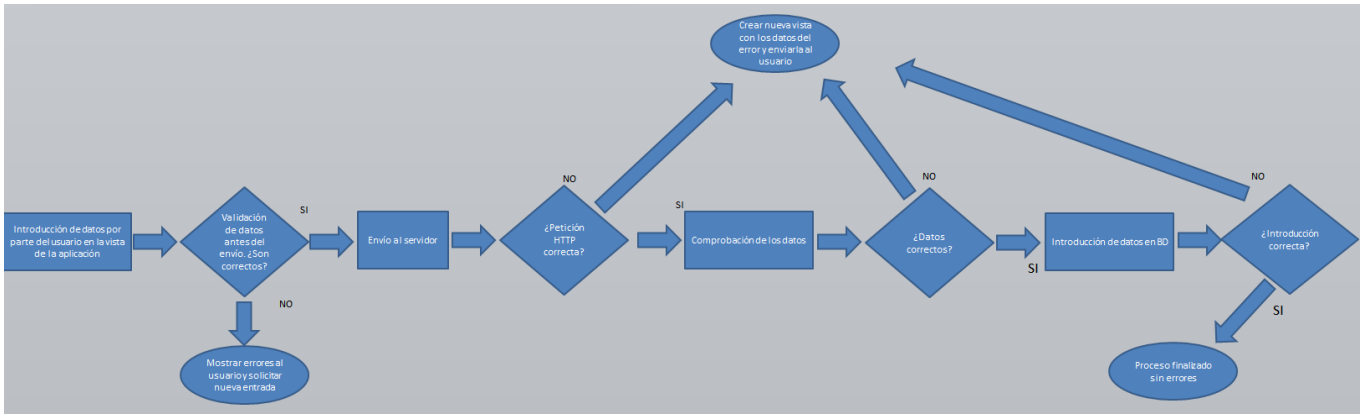

Ilustración 13 Puntos sensibles a errores.

Clase ExcepcionModuloEdicion

Esta clase hereda de la clase Exception de PHP, el modelo de excepciones en PHP es similar al de Java. Cuando la ejecución de un bloque encuentra un error, se lanza un excepción (throw) la cual puede ser atrapada.

En el proyecto, tras lanzarse una excepción del tipo **ExcepcionModuloEdicion** la clase se encarga de generar la información que se mostrará al usuario, una vez generada esta información se puede pedir en el programa principal que la clase llame a la vista usada para mostrar los errores.

4.3 Validación de datos de entrada

Toda aplicación es susceptible a errores, por ello, además de evitarlos produciendo un código robusto en la fase de desarrollo, se han de seguir una serie de pautas para que cuando la aplicación se encuentre con unas condiciones inesperadas estas sean controladas y la aplicación pueda seguir su normal funcionamiento.

En un módulo cuya función es añadir y editar contenidos de una aplicación es indispensable la entrada de datos por parte del usuario, lo que conlleva que sea necesario realizar un hincapié en el control y la validación de esos datos de entrada.

En el caso concreto de este proyecto, la introducción de datos por parte del usuario se realiza a partir de un formulario HTML y estos datos se envían al controlador PHP correspondiente que se encarga de gestionarlos y añadirlos a la base de datos una vez comprobados. Existen dos puntos en los que se puede comprobar la validez de estos datos introducidos por el usuario, antes de enviarlos al controlador PHP, es decir en el propio formulario HTML y también es posible comprobarlos antes de introducirlos en la base de datos, en el controlador una vez enviados. Dado que el módulo de edición de contenidos es un punto crítico y necesario de la aplicación se ha optado por implementar medidas de control y validación en ambos puntos.

Habitualmente la validación del lado del cliente se realiza para poder dar información al usuario acerca de si los datos introducidos son válidos, mientras que la validación del servidor protege a la aplicación de posibles datos peligrosos que hayan sido introducidos de forma premeditada por el usuario saltándose las validaciones del lado del cliente, con la intención de desestabilizar el sistema.

4.3.1 Validación de datos del lado del cliente

Gracias a las mejoras de HTML5 y con la ayuda de JavaScript es posible realizar una validación de los datos introducidos en el propio navegador del cliente para cada uno de los campos de un formulario. La principal ventaja de este método es que no es necesario enviar todos los datos al servidor para su comprobación, y en caso de que algún dato sea incorrecto el usuario será notificado y podrá corregirlo antes de enviarlo.
Mediante estas comprobaciones es posible detectar tres tipos de fallos en los datos, error de tipos de datos, errores por campos obligatorios vacíos y errores de formato en los campos.

Comprobación de tipo de datos
Se produce cuando una de las entradas recibe un tipo de dato no esperado, por ejemplo cuando en un campo numérico se introducen caracteres. Hay dos formas de mitigarlos, o bien dando el campo como no válido y pidiendo al usuario que reintroduzca el mismo, o ignorando los caracteres no válidos. En este proyecto se ha optado por la primera opción y en el caso de que el tipo de datos no corresponda con el requerido (que es posible consultar en la Ilustración 14 Esquema de validacion de los datos del formulario) será validado como incorrecto y se pedirá al usuario que reintroduzca los datos del campo.
Esta comprobación se puede realizar mediante HTML especificando el tipo de dato con el atributo *type=" [tipo] ".*

Donde [tipo] puede ser, entre otros, alguno de los incluidos a continuación (12):

- **text**: admite cualquier carácter. (sin validación).
- **number**: campo numérico, se pueden establecer restricciones adicionales como el máximo admitido o el número de cifras decimales.
- **date**: fecha, en el formato local del usuario.

Mediante código JavaScript la forma de validar los datos es mediante la ejecución de una función, a la hora de enviar el formulario, que comprueba uno a uno cada dato y en caso de que uno o más no correspondan con el tipo esperado, se muestra un error al usuario y el formulario no es enviado. En el formulario principal de este proyecto solo existen tres tipos de datos para la entrada del usuario: texto, números enteros y valores boléanos.

Comprobación de campos requeridos con datos

Este error se produce cuando alguno de los campos necesarios (generalmente los que se usan como índices en la base de datos) está en blanco. La comprobación en HTML se realiza añadiendo el atributo *required* a los campos obligatorios y mediante JavaScript comprobando si el campo está vacío. Solo existen dos campos obligatorios, el número de Unidad y el número de Página, aunque en el caso de que se añada una pregunta con respuestas una de ellas ha de ser marcada como correcta.

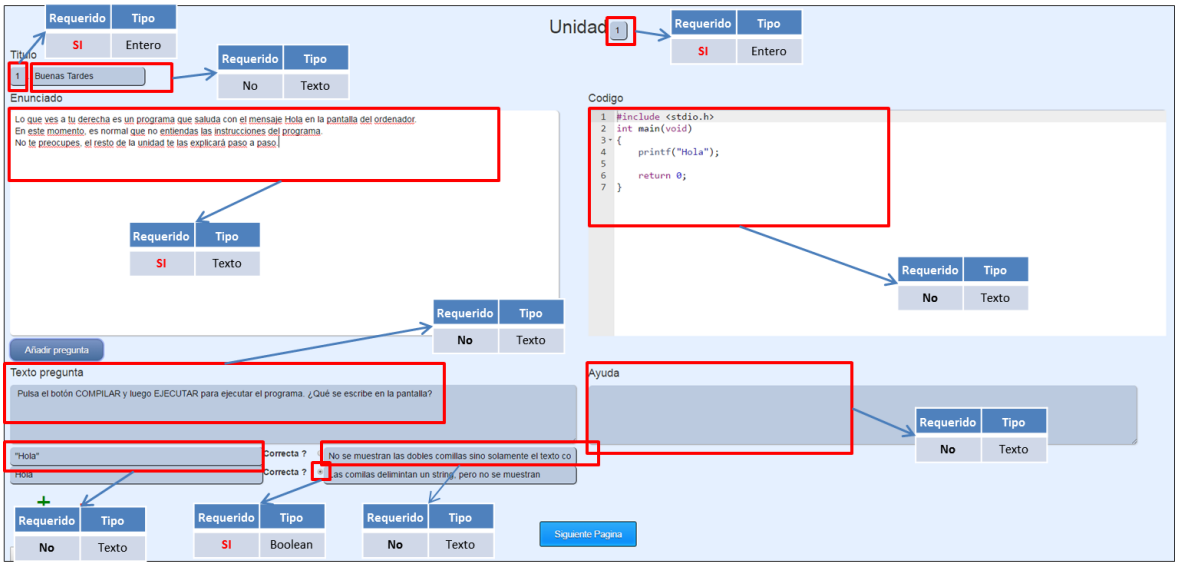

Ilustración 14 Esquema de validacion de los datos del formulario.

La mayor desventaja de este sistema de detección de errores proviene del hecho de que al ser comprobaciones que han sido ejecutadas mediante código en el lado del cliente es imposible asegurar que estas no han sido modificadas de alguna manera y por tanto no se puede garantizar que los datos han sido validados correctamente, por ello, son necesarias comprobaciones del lado del servidor.

4.3.2 Validación de datos del lado del servidor

Una vez que los datos han sido enviados al servidor, debemos comprobar que estos efectivamente siguen el formato esperado y que estos pueden ser introducidos o modificados en la base de datos sin ocasionar problemas.

Comprobación de tipo de datos
Este tipo de control de errores en el lado del servidor se realiza de manera similar a las comprobaciones del lado del cliente, la única diferencia notable es a la hora de informar al usuario de que ha ocurrido un error se lanza una excepción (explicada en el apartado 0), con el tipo de error correspondiente para mostrar al usuario la información relacionada con el error.

Comprobación de datos duplicados / ya existentes
A la hora de introducir nuevo contenido en la base de datos es necesario comprobar si entran en conflicto con datos ya almacenados con anterioridad ya que en caso de que existan datos con la misma clave primaria no será posible introducirlos.
En cambio a la hora de editar el contenido ha de comprobarse justamente el caso contrario, que los datos que se están editando ya existan en la BD.
Para realizar estas comprobaciones, antes de insertar los datos se realiza una consulta usando esa misma clave primaria.

4.4 Autentificación y Seguridad

Gracias al uso framework Laravel el proceso de autentificación de usuarios es prácticamente automático, el framework incluye un módulo que se encarga de solicitar los credenciales de inicio de sesión al usuario, más tarde es posible comprobar si un usuario se encuentra logeado mediante una sencilla función. (13)
Un proceso similar se sigue para comprobar los permisos de acceso de cada uno de los usuarios. En la BD los usuarios tienen un rol asignado (generalmente estudiante o admin). Con respecto al desarrollo del nuevo módulo de edición de unidades, el nivel de acceso mínimo requerido por un usuario para crear o editar unidades debe ser el de administrador de la aplicación.

5 Integración, pruebas y resultados

Toda la fase de diseño y desarrollo del proyecto se ha llevado a cabo en un servidor paralelo distinto al que corre la versión "live" de la aplicación. Hay que destacar que para asegurar una mayor portabilidad del código, el servidor live es totalmente distinto al servidor usado para debug, testing y desarrollo, lo cual garantiza que si todas las pruebas son satisfactorias la aplicación continuará funcionando incluso cambiando la plataforma bajo la que se ejecuta.
En ambos servidores, para llevar un control de versiones y poder gestionar cambios en el código de manera sencilla se ha usado la herramienta git

A continuación se exponen brevemente las características de cada uno de los servidores.

	Servidor live	Servidor desarrollo
HTTP Server	Apache	Nginx
Sistema Operativo	Windows 7	Ubuntu 16.04
PHP Version	PHP 7	PHP 7
Compilador C	MinGW (gcc)	gcc
Servidor Base de datos	MySQL	MySQL

5.1 Pruebas

El framework Laravel pone a disposición del desarrollador la capacidad de crear pruebas unitarias para comprobar el correcto funcionamiento de una aplicación, para ello es necesario crear una clase que es la encargada de contener los métodos necesarios para testear la aplicación. Estos métodos pueden visitar páginas, comprobar el retorno de peticiones HTTP, rellenar formularios, etc. Y es esta última funcionalidad la que se ha usado en este proyecto, ya que la principal fuente de posibles errores es sin duda el formulario que permite editar o añadir nuevo contenido. (14)

Para realizar pruebas con Laravel e interactuar con el formulario se usa un sistema de aserciones, indicando que página se quiere visitar (visit), que campos del formulario se quieren rellenar (type), que salida se espera (see), que página nos encontramos (seePageIs) o que texto no debe aparecer en la misma (dontSee). Cada uno de estos métodos se concatena para que el resultado final de la llamada devuelva un Booleano teniendo valor verdadero en caso de haber superado la prueba.

Un ejemplo de uno de los sencillos tests dentro de esta clase sería el siguiente:

```
public function test_creacion_unidad()
{
    $user = factory(App\User::class, 'admin')->make();
       $this
        ->actingAs($user)
        ->visit('nueva')
        ->type('99','unidad')
        ->type('1','pagina')
        ->type('Test','titulo')
        ->type('texto_ppal','texto_ppal')
        ->type('codigo','codigo')
        ->press('Siguiente Pagina')
        ->seePageIs('nueva')
        ->dontSee('Error.');

}
```

Ilustración 15 Ejemplo test con Laravel.

Y a la hora de ejecutarse las pruebas esta función en cuestión comprobaría paso a paso cada uno de los siguientes puntos:

- Primero se crea un usuario de tipo administrador (ya que son los únicos que poseen acceso a esta sección de la aplicación).
- Se visita la página "nueva" que se corresponde con la página que muestra el formulario de creación de una nueva unidad.
- Se rellenan los distintos campos del formulario: Unidad, pagina, titulo, texto_ppal y código.
- Se presiona el botón "Siguiente página" que añade la página a la base de datos y en caso de que no surja ningún error continúa con el formulario para añadir la siguiente página de la unidad.
- Se comprueba si nos encontramos en la página nueva, es decir si no ha cambiado la página (que es el comportamiento esperado en caso de que se cree una página correctamente).
- Se comprueba que no aparezca ningún error en la página.

En caso de que cada uno de estos pasos sea correcto, la prueba está superada.

Para realizar un testeo completo de los nuevos módulos se han creado dos test diferentes para comprobar el correcto funcionamiento tanto a la hora de añadir páginas como a la hora de editarlas.

5.1.1 Tests para el formulario nueva unidad

Se ha creado el fichero NuevaUnidadTest dentro de la carpeta tests de Laravel, este contiene la clase y métodos adecuados para testear el correcto funcionamiento del módulo a la hora de añadir unidades.
Dentro del mismo se encuentran los diferentes métodos para testear las diferentes entradas y la correcta gestión de errores. A continuación se detallan uno a uno los métodos, su funcionamiento y la salida esperada.

Test usuario sin permiso
Tras crearse un usuario sin privilegios de administrador se comprueba que este no pueda acceder al formulario de creación de páginas y que en su lugar aparezca la página de error.

Test acceso
Se comprueba el caso contrario al anterior test, el usuario posee privilegios de administrador y debería poder ver el formulario.

Test datos erróneos
Una vez accedido al formulario se intentan forzar datos erróneos en el mismo, como por ejemplo el número de Unidad a -1 (debe ser mayor de 0 para ser válido).

Test creación unidad
Este test comprueba que es posible crear una página de una unidad con los datos básicos y correctos, tras enviar los datos al servidor, estos deberían estar guardados y la página web devuelta debería ser la de añadir la siguiente página de la unidad.

Test clave duplicada
Se comprueba que no es posible añadir dos páginas con el mismo número de unidad y pagina (estaríamos en un caso de edición que se realiza en una página de la aplicación diferente), para ello si intenta añadir una página idéntica a la del anterior test, que ya debería estar en la base de datos. El resultado esperado debería ser una página de error, y en caso de que este test fallase bien podría ser porque ocurrió un fallo en el anterior y no se insertaron los datos o bien porque existe un error de programación a la hora de controlar la inserción de claves primarias duplicadas, por ello se creó el siguiente test.

Test datos almacenados en BD
Este último test comprueba que los datos que se enviaron a través del formulario a la base de datos existan en la misma. En concreto se comprobará que exista un registro con el número de Unidad 99, página 1 y título test.

Una vez construidos todos los test, se ejecutan desde la consola. Tras realizar pequeñas modificaciones en los mismos para ajustarlos a casos reales el resultado final de correr los test fue el esperado.

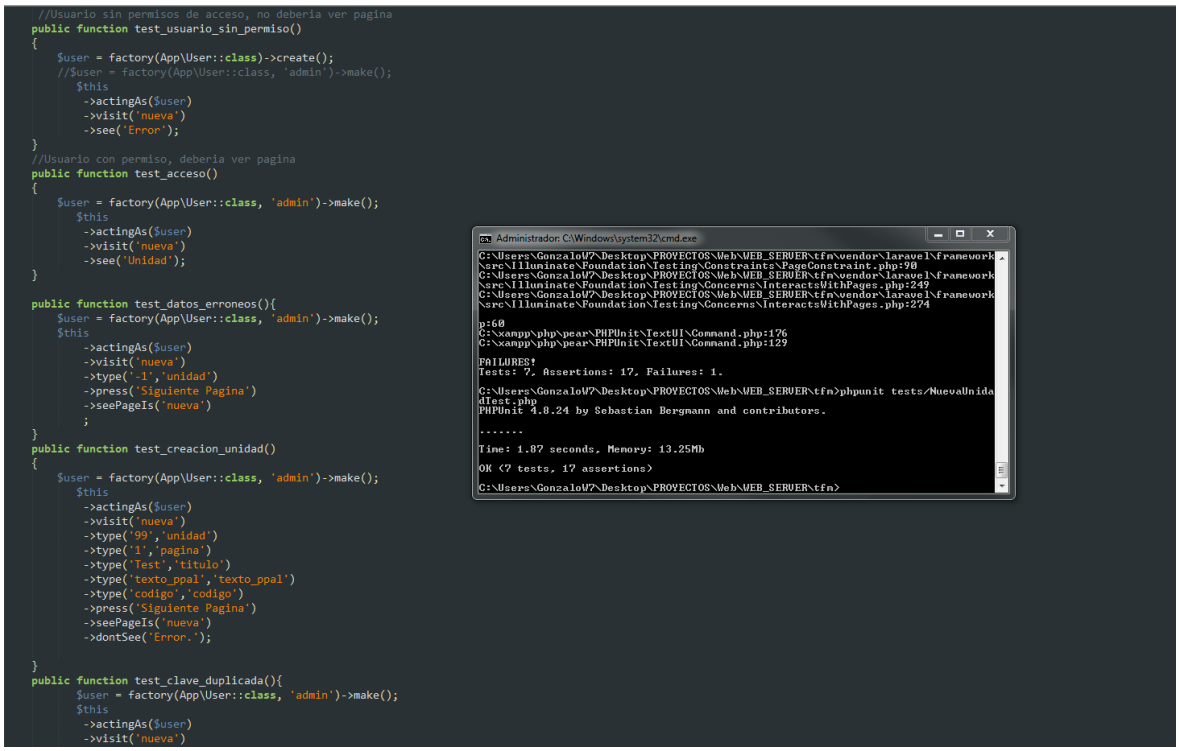

Ilustración 16 Tests y resultados de los mismos.

5.1.2 Tests para el formulario de edición de unidad

Estos tests tienen el objetivo de asegurar el correcto funcionamiento del formulario en modo edición. Algunas comprobaciones como las comprobaciones de claves primarias ya no son necesarias, y otras son semejantes a las realizadas en los tests de creación de unidades, en cambios en su lugar es necesario realizar pruebas para comprobar que los datos son editados correctamente.

Test página correcta

Test básico para comprobar que la página se encuentra disponible y accesible para los perfiles con permiso para visualizarla.

Test error permisos

Comprueba que no es posible acceder a la página sin permisos de administración.

Test datos cargados correctamente

Comprueba que los datos concretos de una unidad y pagina son cargados correctamente de la base de datos.

Test edición unidad

Test para comprobar que es posible y se editan los datos de una unidad mediante el formulario.

Tests datos editados

Se comprueba que los datos fueron modificados en el anterior test volviendo a cargar el formulario, y comprobando que estos son los correctos

Tests datos en base de datos

Se comprueba que los datos también son editados en la base de datos correctamente.

6 Conclusiones y trabajo futuro

Con el desarrollo de este trabajo de fin de grado se ha perseguido el objetivo de diseñar, desarrollar, probar e integrar un nuevo y completo modulo para una herramienta de aprendizaje online, que permitirá a los usuarios, más en concreto al personal docente añadir contenidos de una manera mucho más sencilla, segura y eficiente.

A modo de conclusión final, a continuación se desarrolla una lista con los logros, objetivos cumplidos y la experiencia obtenida mediante la realización de este trabajo.

- Se ha conseguido retomar un proyecto realizado por otra persona. A partir de un estudio de su implementación, las decisiones de diseño tomadas y el propio uso de la aplicación se han podido aplicar las mismas técnicas en la creación del nuevo módulo. Este es un aspecto muy importante y muchas veces olvidado en el desarrollo de software, no solo es importante saber comenzar desde un principio, también es necesario saber analizar aplicaciones ya construidas y mantenerlas o mejorarlas siguiendo los mismos patrones.

- Se han cumplido los requisitos especificados para el funcionamiento independiente de este nuevo módulo, es decir, cualquier cambio en la aplicación base no afectará al funcionamiento de este módulo, siempre y cuando se continúe usando la misma arquitectura y modelo de datos que se usa actualmente.

- El nuevo módulo evita tener que editar el código de la aplicación para modificar su contenido, ahora los datos correspondientes a este contenido se encuentran separados como dicta el patrón de diseño que se ha seguido (MVC), por tanto cualquier modificación de los controladores que cargan estos datos no afectara a los mismos, como ocurría anteriormente.

- Ahora la aplicación permite que esta tarea de inclusión o edición de contenidos pueda realizarse por cualquier persona, sin necesidad de poseer conocimientos de programación ya que esto se realiza a través de un sencillo formulario con una interfaz sencilla y accesible a cualquiera.

- Se ha conseguido construir un módulo robusto y seguro, y esto es gracias en gran parte a la sencillez que presenta Laravel a la hora de construir este tipo de aplicaciones basadas en la autentificación de usuarios.

- Se ha creado un sistema de gestión de errores completo y conciso, de tal manera que cualquier error surgido a la hora de modificar contenidos será fácilmente trazable y por tanto más sencillo de solucionar.

- Se ha diseñado y construido un sistema de pruebas unitarias que permite comprobar el correcto funcionamiento de la aplicación antes de publicarla tras realizar cualquier cambio, es importante destacar que esta parte no existía con anterioridad y gracias al trabajo realizado será posible crear nuevas y distintas pruebas para los futuros módulos y secciones que se desean añadir.

- El modulo es totalmente escalable. En caso de querer realizarse cambios o mejoras futuras en el mismo, será posible hacerlo de una manera sencilla.

- Por último se ha asegurado que cualquier trabajo que se realice posteriormente será fácilmente integrable con cualquiera de las partes desarrolladas a lo largo de este trabajo.

En cuanto a la experiencia obtenida con el desarrollo de este trabajo me gustaría destacar que una de las más importantes ha sido el aprendizaje sobre como estudiar una aplicación existente y a partir de su código deducir la forma en la que fue diseñada, el porqué de las decisiones que se tomaron y más importante, como continuar su desarrollo de tal manera que sea posible que otra persona lo continúe en el futuro.
También es de gran valor la experiencia obtenida en cuanto al uso de frameworks, más en concreto el framework de PHP, Laravel. Es uno de los más usados y reputados a nivel mundial, y sin duda el trabajo realizado con el mismo servirá de ayuda en el futuro a la hora de realizar nuevos proyectos.
Por último también he adquirido conocimientos sobre la gestión de los datos dentro de una aplicación, y como diseñar un sistema que los gestione y permita su modificación de manera adecuada y sin fallos.

La aplicación continuará creciendo, y esto será posible a su diseño modular, pero con respecto a la parte que atañe a este módulo concreto existen ciertas mejoras que sería posible implementar y darían una mejor experiencia de uso al usuario final.
Entre ellas las más interesantes de desarrollar en un futuro cercano serían las siguientes.

Gestión de errores

La gestión de errores se realiza mediante excepciones en concreto mediante la excepción creada y explicada en el apartado 4.2(Control y manejo de errores).
Esto permite que en caso de que ocurra un error este siempre seguirá los mismos “cauces” para reportarlo al usuario. Una mejora podría ser la creación de un pequeño modulo que se encargase de agrupar estos errores de tal manera que fuera más sencilla su gestión por los desarrolladores a cargo de solucionarlos, esto podría hacerse a través de un log o de nuevas tablas en la base de datos.

Interfaz

El uso de dispositivos móviles aumenta constantemente, por lo tanto es muy probable que en el futuro las aplicaciones de aprendizaje online como esta sean mayormente usadas desde los mismos. Por ello una mejora significativa sería un remodelado de la interfaz para adaptarla a dispositivos móviles, haciéndola más accesible.
También se podría realizar mejoras que permitiesen añadir o editar unidades por lotes, en el caso de que se deseara.

Gestión de ejercicios

Actualmente la aplicación solo da soporte a un tipo concreto de ejercicios basados en preguntas y respuestas. Con la creación de un módulo completo encargado de gestionar ejercicios, podrían realizarse algunos más variados y diferentes para cada unidad, para realizar esta mejora se deberían realizar tareas de integración de ese nuevo módulo con el desarrollado para este proyecto. Así mismo una de las fortalezas de esta aplicación es la ejecución de código C en la misma, con la posibilidad de editarlo por parte del alumno, sería interesante una mezcla de estos nuevos ejercicios con esta parte de la aplicación.

Referencias

1. **Menarguez, Ana Torres.** EL Pais. [En línea] 26 de 11 de 2014. http://economia.elpais.com/economia/2014/11/25/actualidad/1416928825_955701.html.
2. **Ellis, Ryann K.** *A field Guide to Learning Management Systems.*
3. **eLearning Industry.** Choosing an online learning platform. [En línea] 2017. https://elearningindustry.com/choosing-online-learning-platform-makes-sense.
4. **Tiobe.** Tiobe. [En línea] 11 de 2016. http://www.tiobe.com/tiobe-index//.
5. **ECMA International.** ECMA. [En línea] 2016. http://www.ecma-international.org/memento/history.htm.
6. **The PHP Group.** PHP Manual. [En línea] 2016. http://php.net/manual/en/intro-whatis.php.
7. **Otwell, Taylor.** About Laravel. [En línea] 2016. https://github.com/laravel/laravel/blob/master/readme.md.
8. **Wikipedia - SQL.** SQL - Wikipedia. [En línea] 12 de 2016. https://es.wikipedia.org/wiki/SQL.
9. **United States Department of Health and Human Services (HHS).** Selecting a development approach. [En línea] 2008. https://www.cms.gov/Research-Statistics-Data-and-Systems/CMS-Information-Technology/XLC/Downloads/SelectingDevelopmentApproach.pdf.
10. **Boehm, Barry.** A spiral Model of Software Development and Enhancement. [En línea] 1986. http://dl.acm.org/citation.cfm?doid=12944.12948.
11. **Cockburn, Dr. Alistair.** Using Both Incremental and Iterative Development. [En línea] 2008. http://static1.1.sqspcdn.com/static/f/702523/9242211/1288741989673/200805-Cockburn.pdf?token=PpeQuRbFfeQmQaGI0oaA9Ow0%2Bho%3D.
12. **W3Schools.** HTML5 input types. [En línea] http://www.w3schools.com/html/html_form_input_types.asp.
13. **Laravel.** Laravel - Security. [En línea] https://laravel.com/docs/4.2/security.
14. —. Testing. [En línea] https://laravel.com/docs/5.1/testing.
15. **Ruano, Pablo Molins.** *Desarrollo de un sistema de cuestionarios adaptativos para el apoyo al aprendizaje.* 2015.
16. **Mingorance, Javier Rubio.** *Diseño de videojuegos para el aprendizaje.*
17. **Gamma, Erich.** *Design Patterns.* 2008.
18. **Bucher, Taina.** *Objects of Intense Feeling : The case of the Twitter API.*
19. **Wikipedia.** Modelo-Vista-Controlador Wikipedia. [En línea] 14 de 10 de 2016. https://es.wikipedia.org/wiki/Modelo%E2%80%93vista%E2%80%93controlador.
20. —. Sistema de Gestión de Aprendizaje - Wikipedia. [En línea] 26 de 10 de 2016. https://es.wikipedia.org/wiki/Sistema_de_gesti%C3%B3n_de_aprendizaje.
21. **Otwell, Taylor.** Documentacion Laravel. [En línea] 2016. https://laravel.com/docs/5.3.

Glosario

API	Application Programming Interface. Conjunto de herramientas que un software ofrece, para facilitar el desarrollo de otros programas que realizan peticiones al mismo.
compilador	Herramienta que permite traducir el código escrito en un lenguaje de programación a lenguaje máquina para su ejecución en la CPU.
Feedback	En el entorno de una aplicación, recoger reacciones de los usuarios para modificar el funcionamiento de la misma de acuerdo con sus solicitudes.
Framework	Estructura base que permite construir proyectos de software usando una serie de librerías, bibliotecas y funciones que sirven de soporte para el código escrito por el usuario.
Laravel	Framework de código abierto desarrollado en PHP usado en el desarrollo de aplicaciones web siguiendo el modelo vista-controlador (MVC).
Lenguaje de marcas	Forma de codificar un documento incorporando junto con el contenido etiquetas o marcas que contienen información acerca de su estructura o presentación.
Lenguaje de programación interpretado	Lenguaje el cual no es necesario compilar para su ejecución, sino que se traduce a lenguaje máquina parcialmente, conforme se va ejecutando.
MVC (Modelo Vista Controlador)	Patrón de diseño de arquitectura de software que separa los datos y la lógica de una aplicación de la interfaz de usuario (19)
PHP	Lenguaje de programación del lado del servidor (Server-side) concebido principalmente para el desarrollo web. Es un lenguaje interpretado.
Server-side	Cuando se utiliza refiriéndose a código, significa que este se ejecuta en el servidor, y la salida de esta ejecución es enviada al usuario.
Wiki	Sistema de páginas web cuyos contenidos son editables por los mismos usuarios y en las que se presenta información a modo de enciclopedia virtual
World Wide Web	Red informática de distribución de información entre los dispositivos conectados a la misma.

Anexos

A Códigos de error del módulo de Excepciones

Para una mayor facilidad a la hora de comprender los errores, se ha decidido que para la gestión de los mismos se usen códigos o etiquetas descriptivas.
De esta manera cuando se lanza una excepción ExcepcionModuloEdicion se debe indicar el código de error (si procede) que se ha producido. Actualmente es posible usar cualquiera de los códigos de error mostrados a continuación.

CLAVE_DUPLICADA
Se produce cuando se intentan añadir nuevo contenido a la Base de Datos, pero ya existen datos que comparten la misma clave primaria que los que se intentan añadir, este tipo de error, actualmente, solo puede darse a la hora de crear nuevas Unidades, ya que es el único momento que puede existir un conflicto de claves primarias.

DATOS_NO_EXISTENTES
Tipo de error análogo al anterior, pero en este caso, el error es producido a la hora de intentar editar datos ya almacenados en la BD y al intentar acceder a ellos por sus claves primarias estos no se encuentren ya en la base de datos.

PERMISOS_ACCESO
Se produce cuando un usuario sin los permisos adecuados intenta acceder a una parte de la aplicación donde se requiere un rol con más privilegios.

ERROR_EN_TIPO_DATOS
Cuando se reciben unos datos de entrada (generalmente de un formulario rellenado por el usuario) distintos de los esperados, se lanza este tipo de error. Por ejemplo, cuando se recibe un carácter cuando se esperaba un número.

ERROR_DATOS_FORMULARIO
Otro error distinto que puede ser lanzado a la hora de procesar datos provenientes de formularios, este error en concreto indica que algún parámetro de la petición del formulario contiene caracteres incorrectos o no es posible almacenarlo en la BD.

B Manual de usuario

A continuación se presenta un pequeño manual de uso que explica cómo se puede acceder y utilizar la nueva funcionalidad.

- El primer paso es logearse con una cuenta de administrador, ya que son las únicas que tienen permiso a estos módulos.

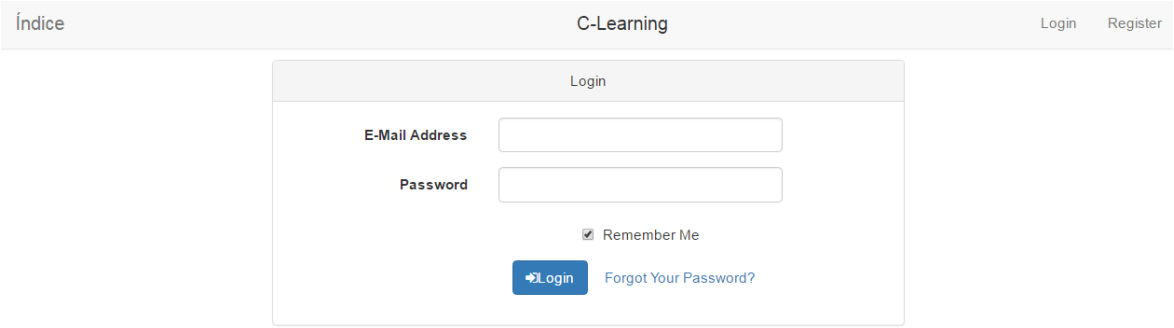

Ilustración 17 Pantalla de Login

- Una vez logeados el programa nos redireccionará a la pantalla del panel de administración, donde a parte de las estadísticas podremos seleccionar una de las nuevas opciones: Nueva unidad / Edición.

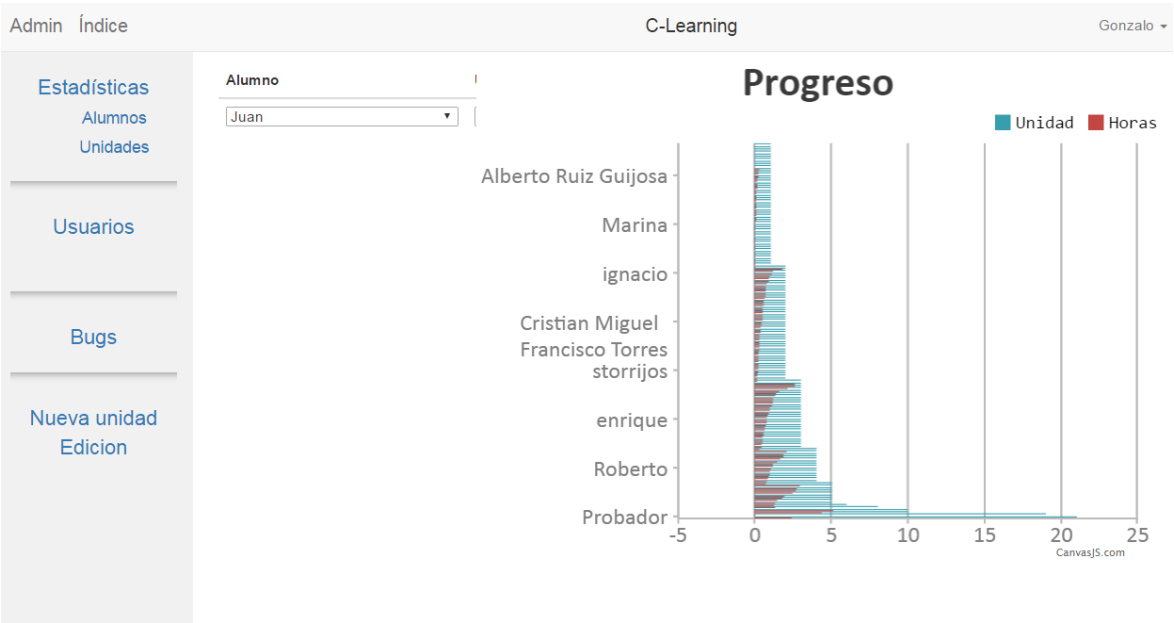

Ilustración 18 Panel de administración

- Si seleccionamos la opción Nueva unidad podremos acceder al formulario de creación de unidades.

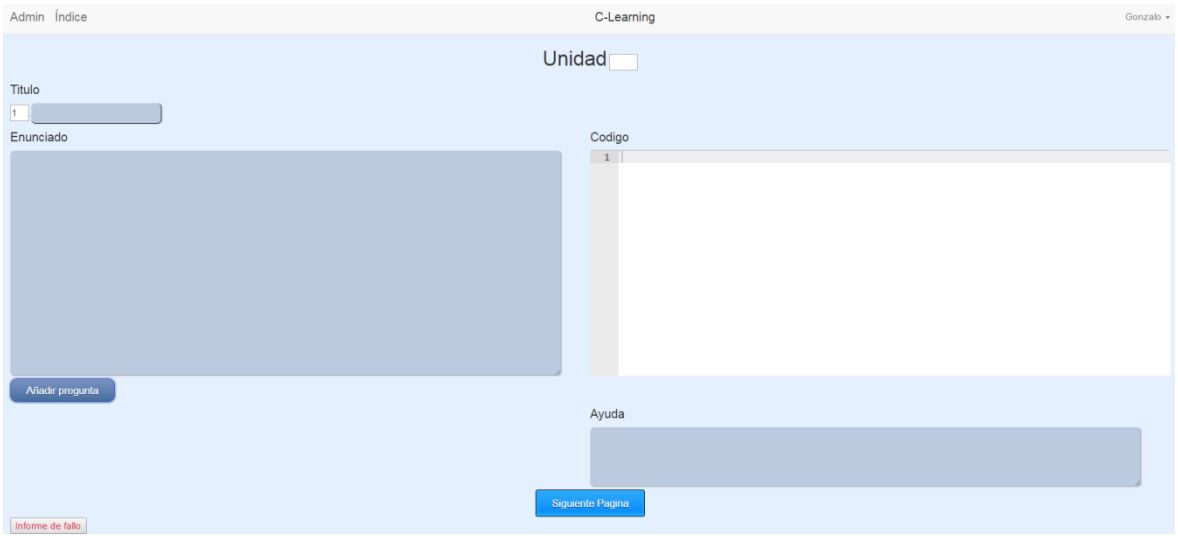

Ilustración 19 Formulario de creación de unidades

- Si deseamos añadir una pregunta para que el alumno responda, debemos hacer clic en el botón Nueva pregunta, lo que desplegara los siguientes campos.

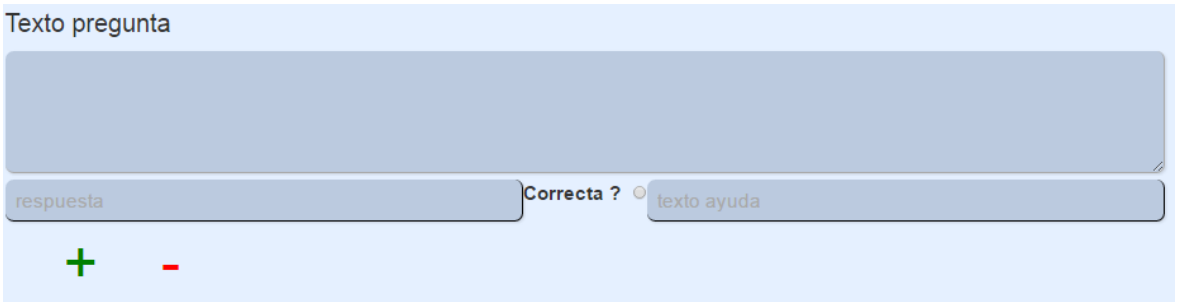

Ilustración 20 Formulario para insertar preguntas en una página

- Aquí tras introducir el texto de la pregunta, podemos añadir respuestas con los botones "+" o "-"y seleccionar la correcta mediante el radio button, así como incluir un texto de ayuda.

- Para guardar los cambios simplemente debemos hacer clic en el botón Siguiente página, si no aparece ningún error por la pantalla y se nos presenta de nuevo el formulario para añadir la siguiente página de la unidad, los datos se habrán guardado correctamente.

- Por último si deseamos editar una unidad ya guardada, debemos volver al panel de administración y seleccionar “Editar”. Una vez hecho esto, aparecerá una lista con las unidades ya guardadas en la aplicación, donde podremos desplegar una y seleccionar la página a editar.

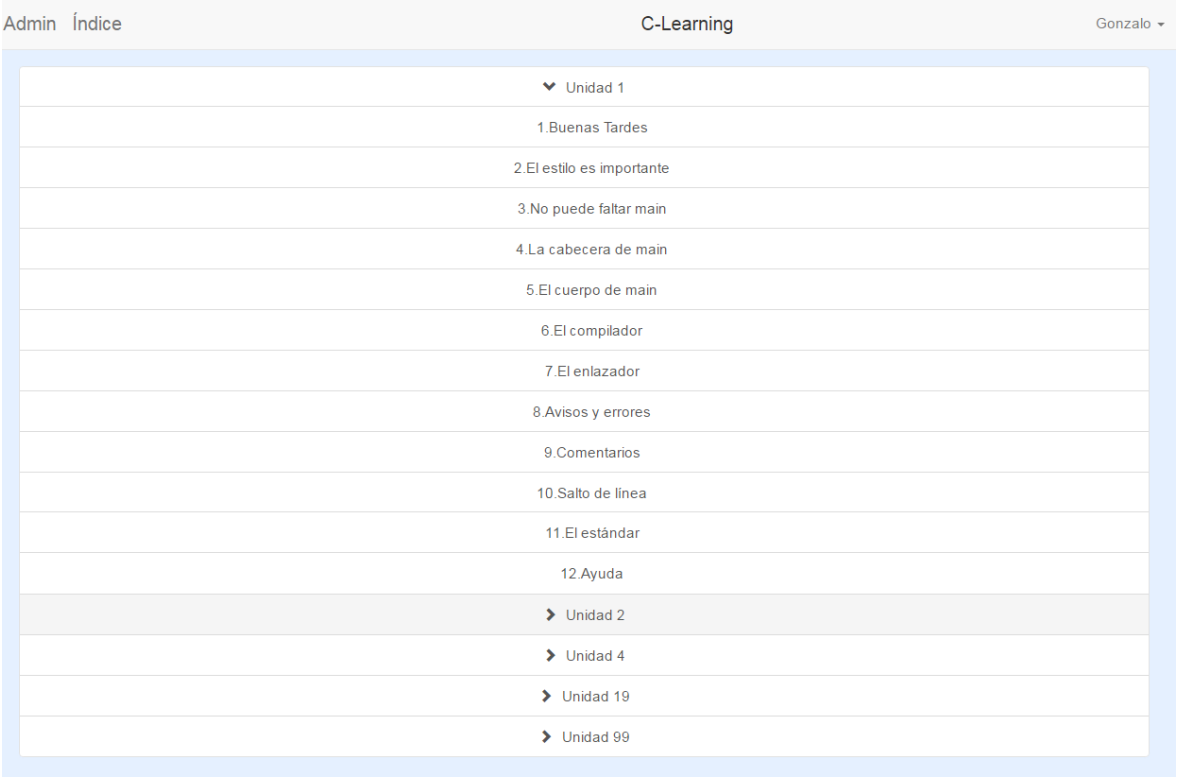

Ilustración 21 Menu de unidades desplegado

- Una vez seleccionada se nos presentará el mismo formulario, con los datos adecuados cargados, el cual podemos editar y guardar los cambios como anteriormente.

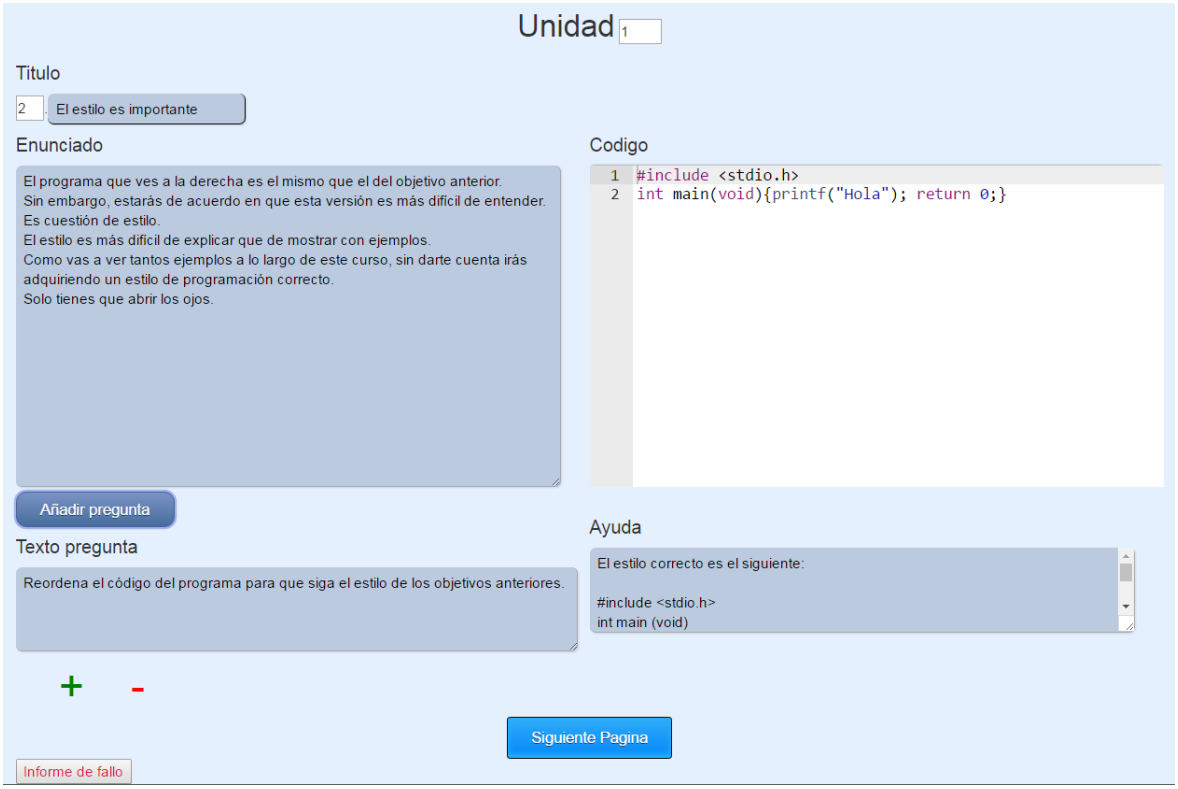

Ilustración 22 Formulario edicion con los datos cargados.